Open Talk 公开课

Traefik 在又拍云的应用和改造

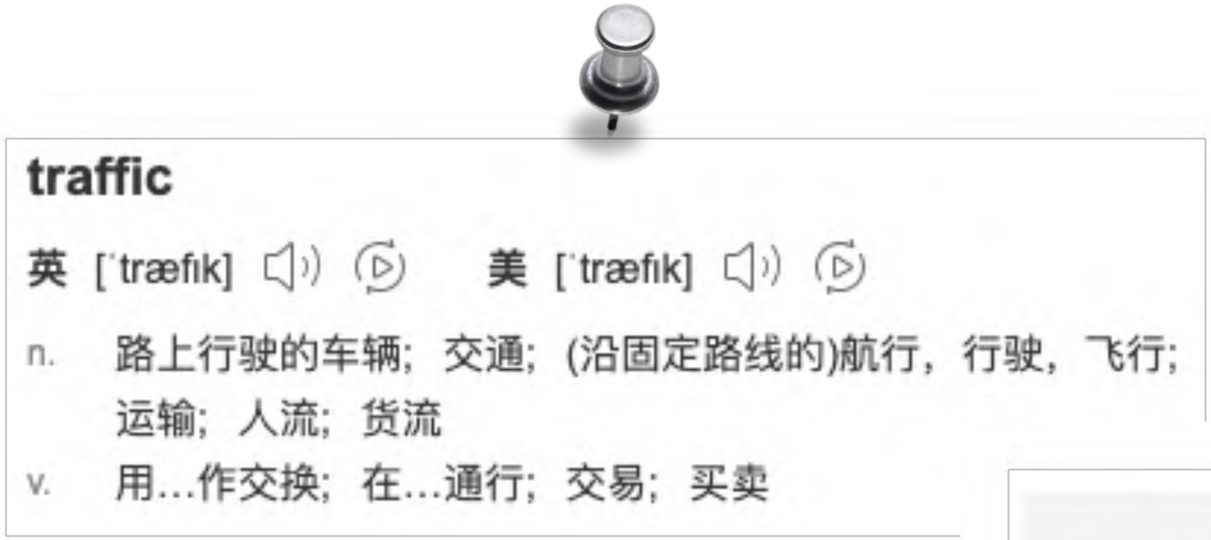
陈卓 又拍云系统开发工程师

又拍云

分享内容

- Traefik 简介
- Traefik 跟 Ingress-Nginx 比较
- 我们为什么使用 Traefik
- Traefik 改造之路

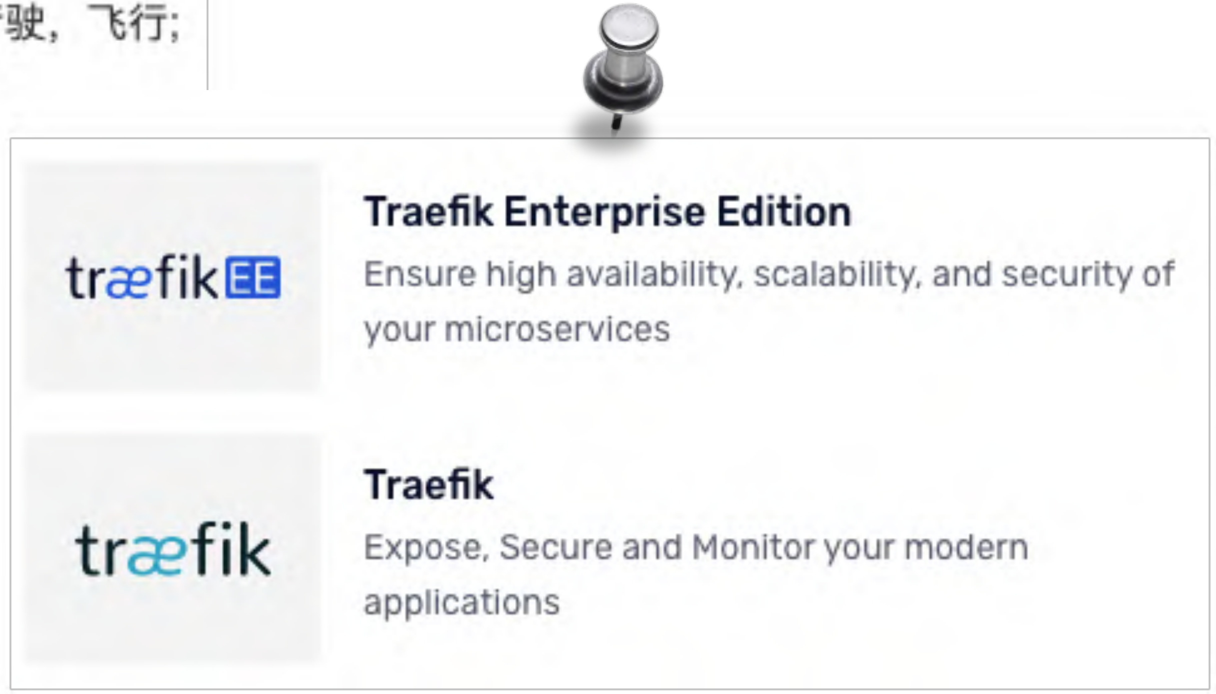
Traefik 简介



traffic

英 [ˈtræfɪk] 美 [ˈtræfɪk]

n.	路上行驶的车辆；交通；(沿固定路线的)航行，行驶，飞行；运输；人流；货流
v.	用…作交换；在…通行；交易；买卖



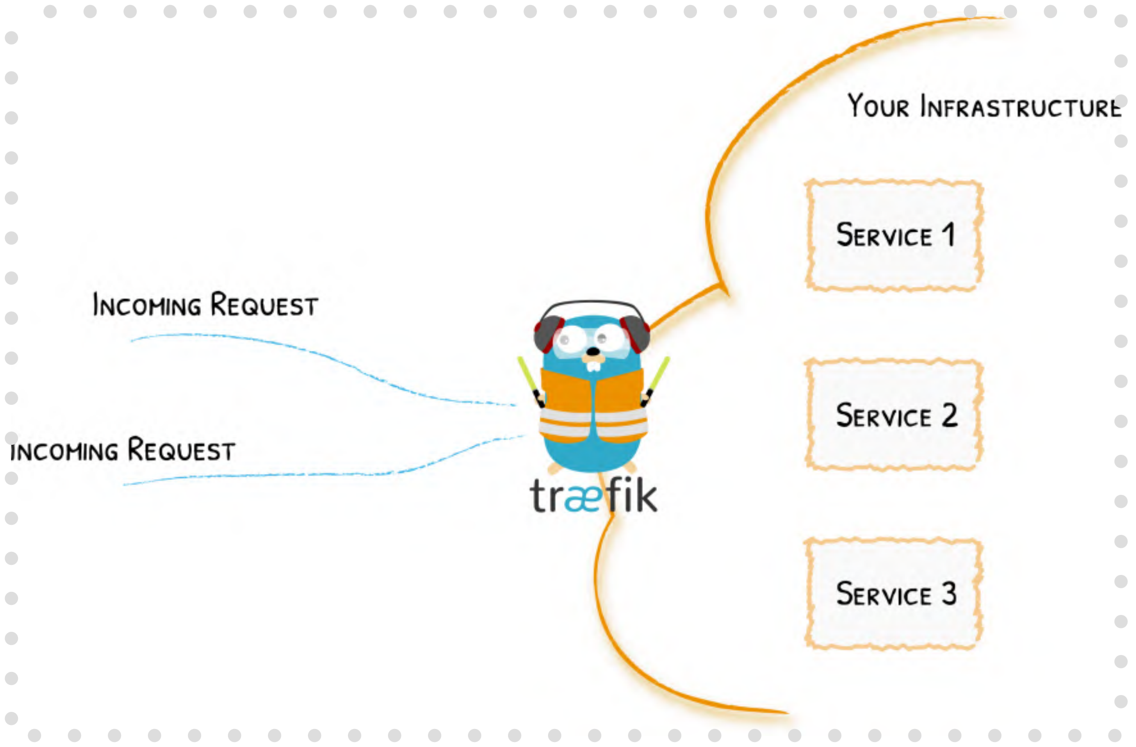
Traefik Enterprise Edition

Ensure high availability, scalability, and security of your microservices

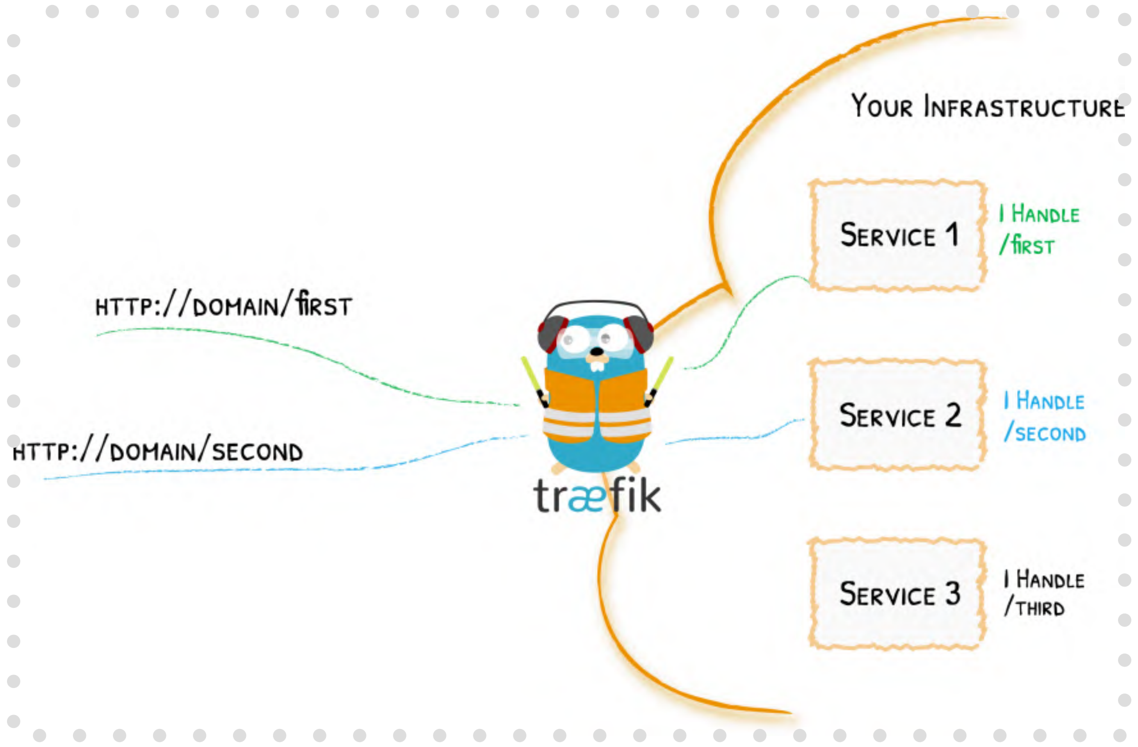
Traefik

Expose, Secure and Monitor your modern applications

Traefik 简介

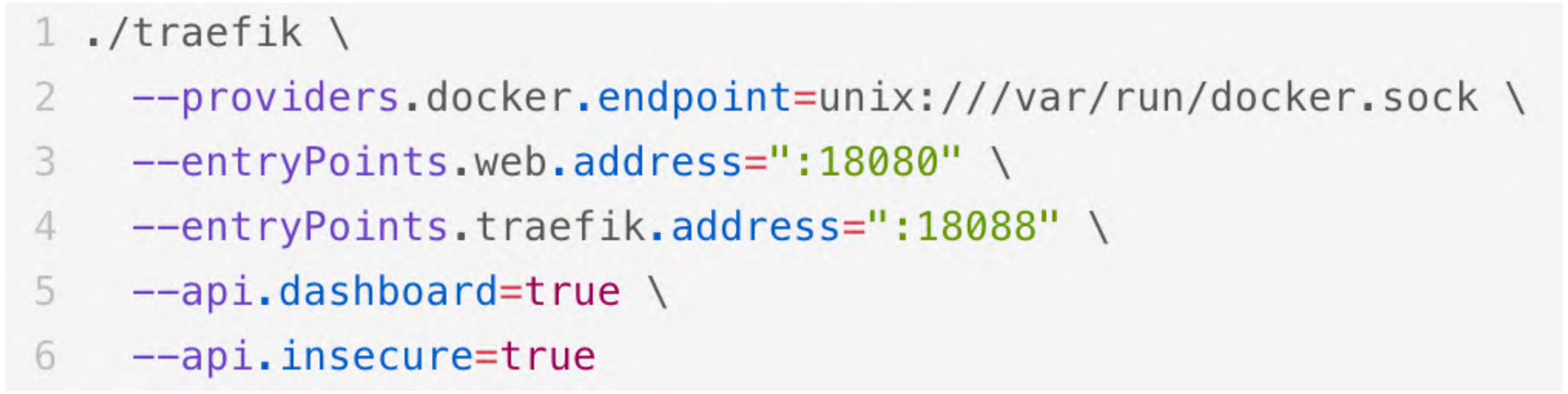


Edge Router



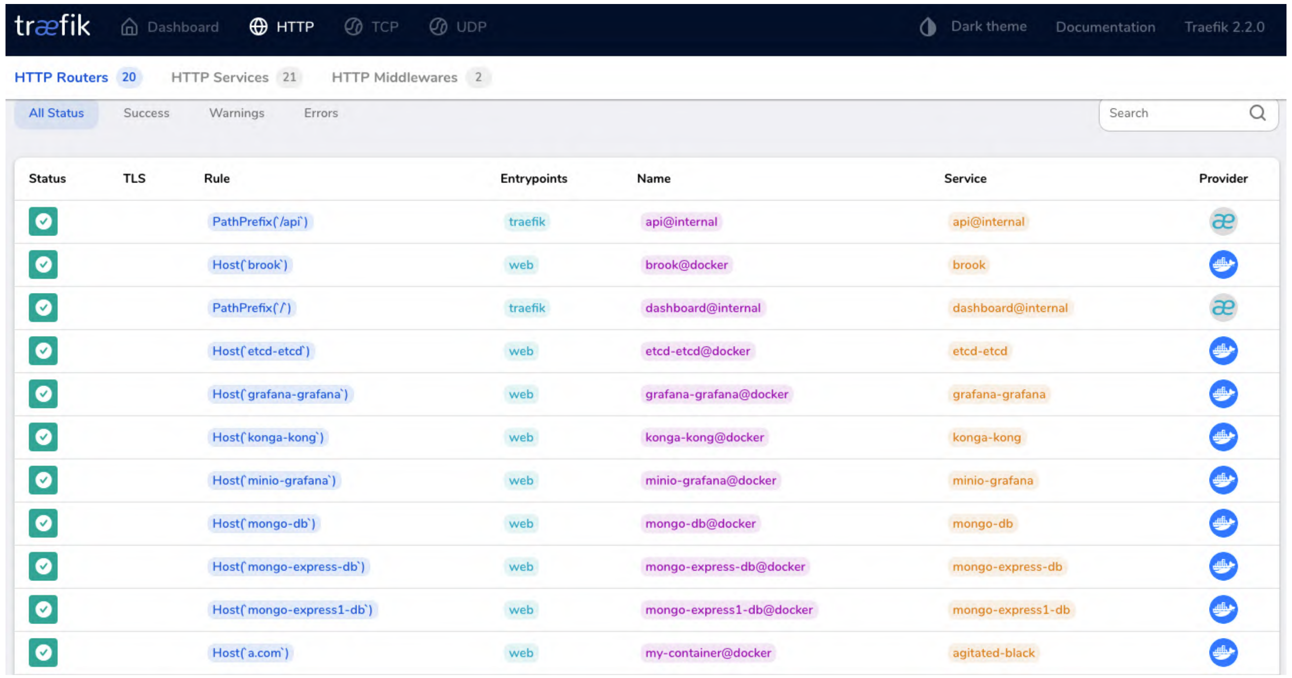
Auto Service Discovery

Traefik 简介



```
./traefik \
  --providers.docker.endpoint=unix:///var/run/docker.sock \
  --entryPoints.web.address=":18080" \
  --entryPoints.traefik.address=":18088" \
  --api.dashboard=true \
  --api.insecure=true
```

Traefik 简介

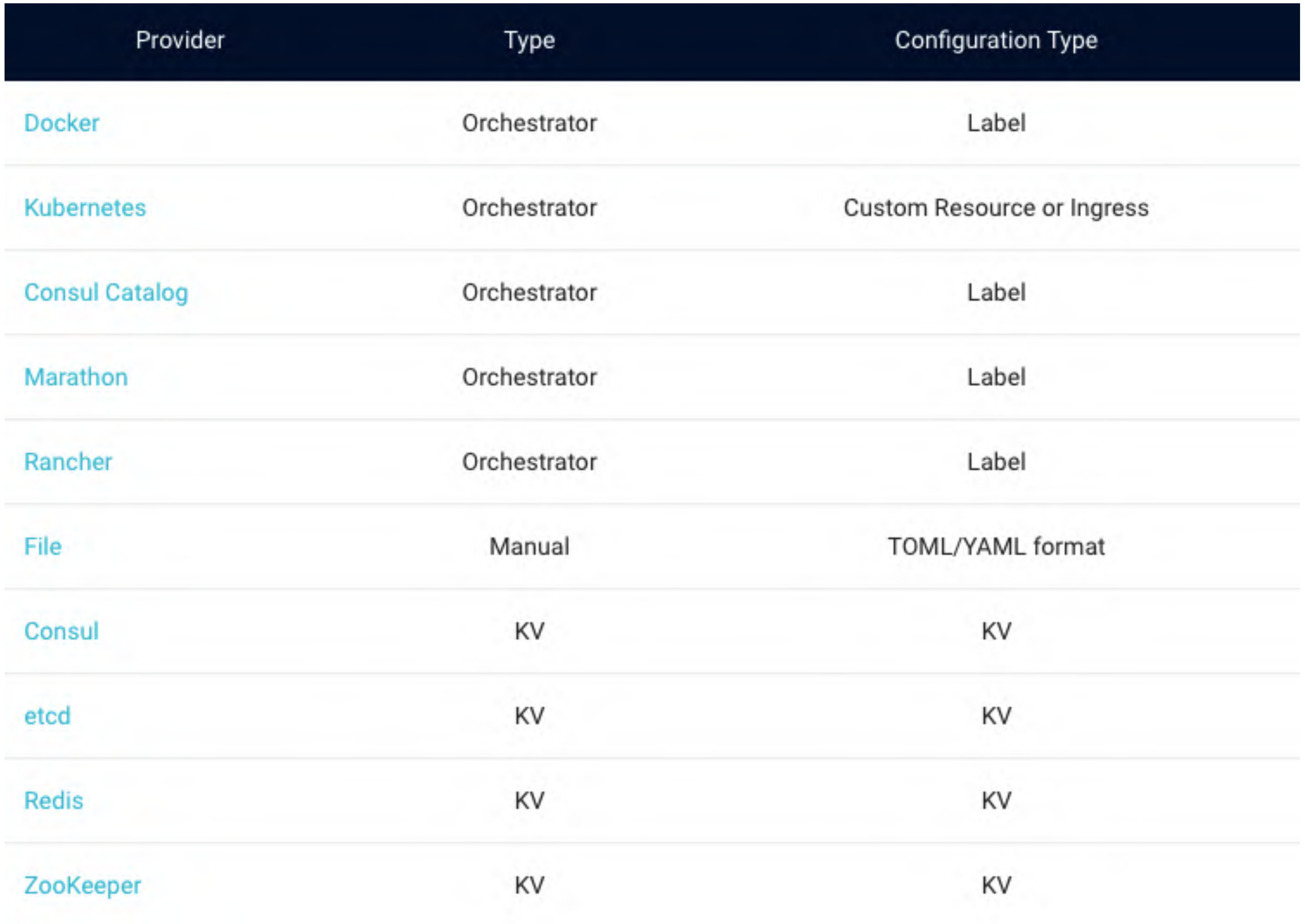


Status	TLS	Rule	Entrypoints	Name	Service	Provider
✓		PathPrefix(`/api`)	traefik	api@internal	api@internal	
✓		Host(`brook`)	web	brook@docker	brook	
✓		PathPrefix(`/`)	traefik	dashboard@internal	dashboard@internal	
✓		Host(`etcd-etcd`)	web	etcd-etcd@docker	etcd-etcd	
✓		Host(`grafana-grafana`)	web	grafana-grafana@docker	grafana-grafana	
✓		Host(`konga-kong`)	web	konga-kong@docker	konga-kong	
✓		Host(`minio-grafana`)	web	minio-grafana@docker	minio-grafana	
✓		Host(`mongo-db`)	web	mongo-db@docker	mongo-db	
✓		Host(`mongo-express-db`)	web	mongo-express-db@docker	mongo-express-db	
✓		Host(`mongo-express1-db`)	web	mongo-express1-db@docker	mongo-express1-db	
✓		Host(`a.com`)	web	my-container@docker	agitated-black	

http://127.0.0.1:18088/dashboard/#/http/routers

```
docker run \
  --rm \
  -l "traefik.http.routers.my-container.rule:Host(\`a.com\`)" \
  nginx
```

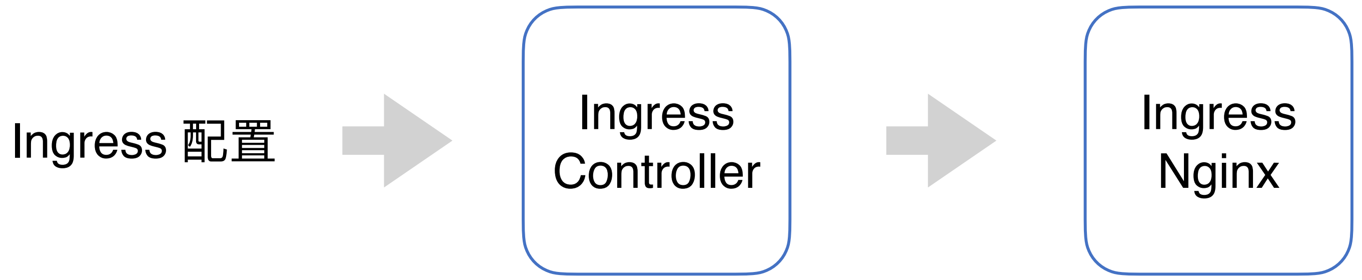
Traefik 配置提供者 —Provider



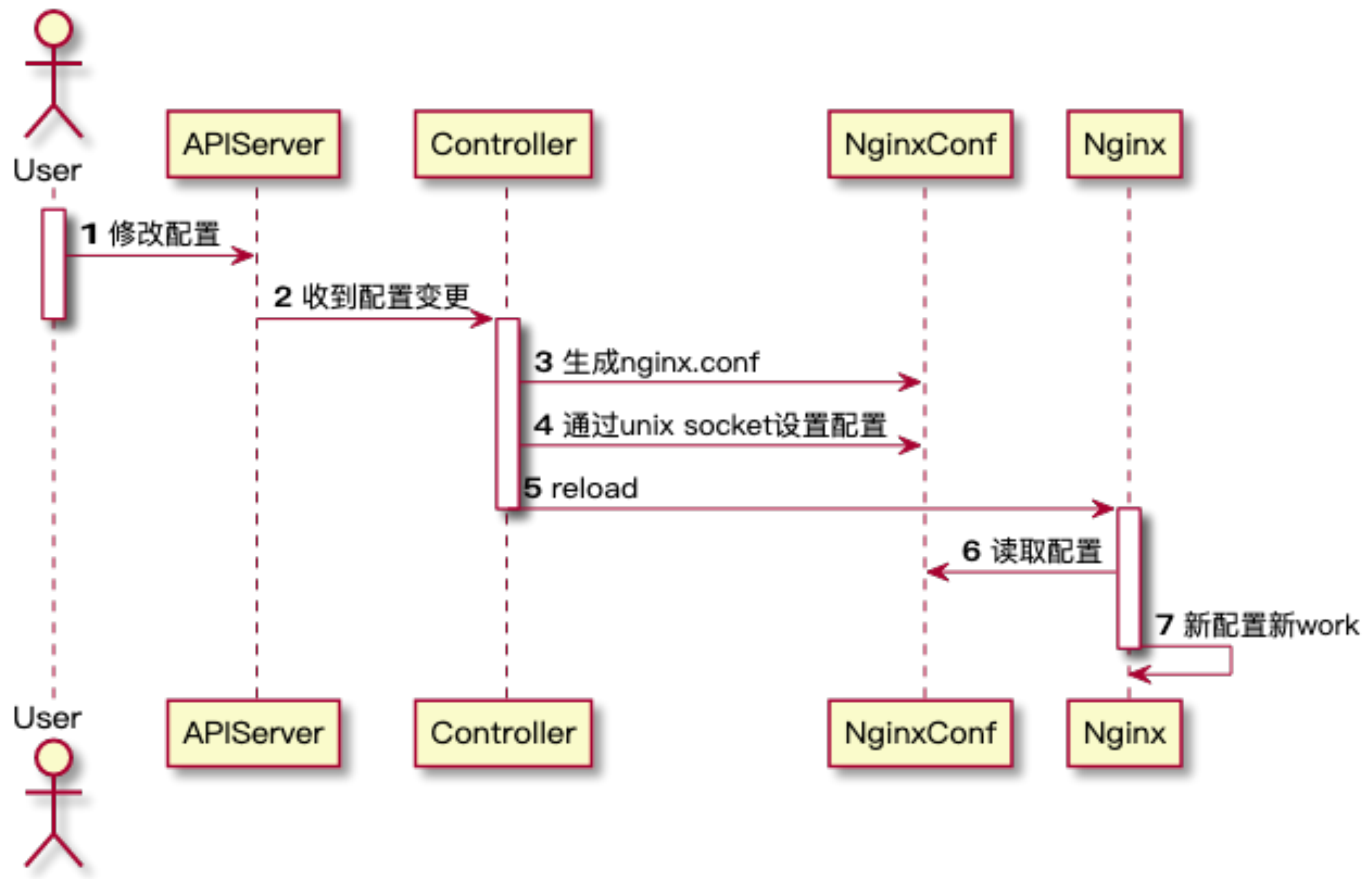
Provider	Type	Configuration Type
Docker	Orchestrator	Label
Kubernetes	Orchestrator	Custom Resource or Ingress
Consul Catalog	Orchestrator	Label
Marathon	Orchestrator	Label
Rancher	Orchestrator	Label
File	Manual	TOML/YAML format
Consul	KV	KV
etcd	KV	KV
Redis	KV	KV
ZooKeeper	KV	KV

Ingress-Nginx 介绍

- Ingress-Nginx：K8S 官方的 Http 网关产品
- Ingress 配置： 指的是 K8S 的 Ingress 的 configmap



Ingress-Nignx 流程

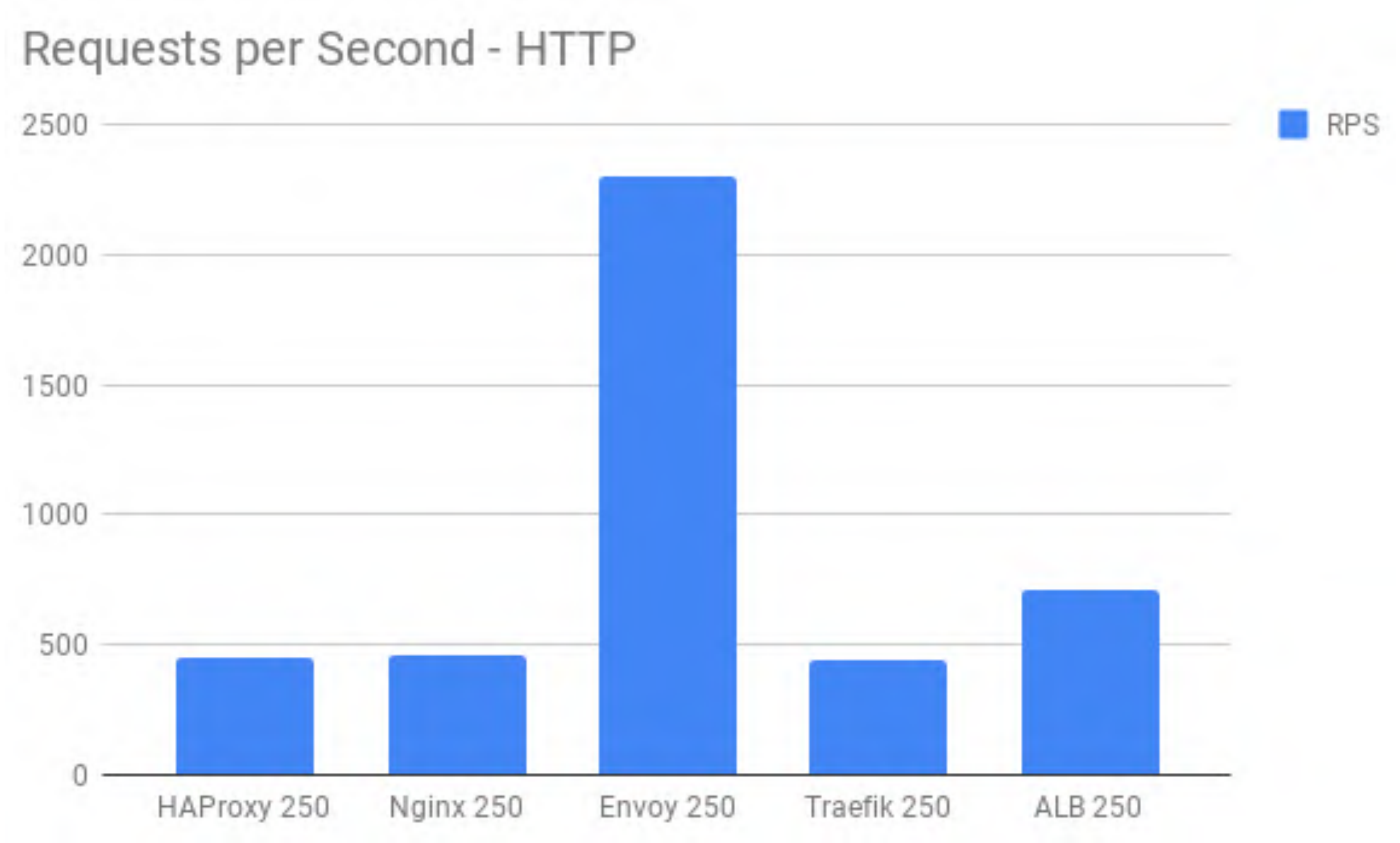


为什么选择 Traefik，不用其它产品

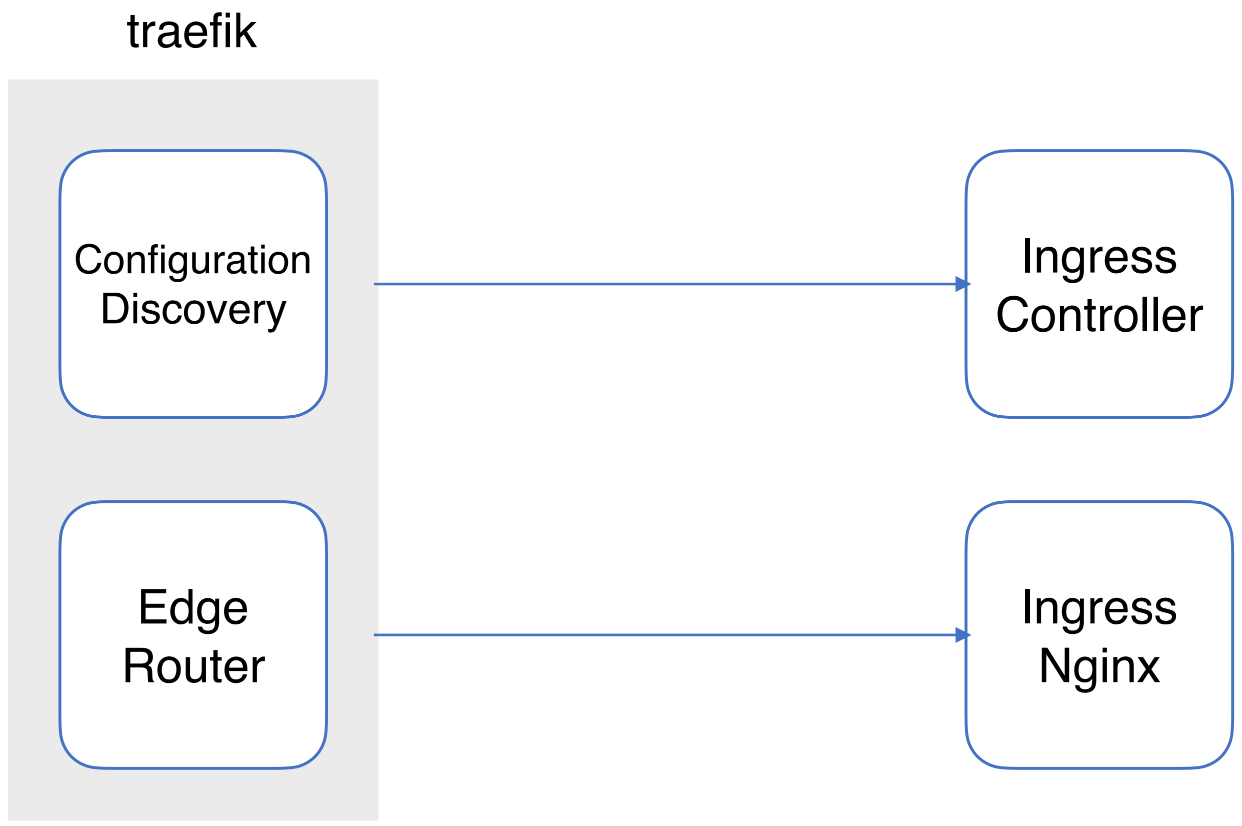
ingress-nginx/kong/apisix	controller 使用 go，网关使用基于 openresty 的软件 性能有保障，但增加修改，kong/apisix 需要额外的存储
envoy/getambassador	envoy 成熟，但是 c++ 的 controller getambassador 使用度不高
traefik	纯 go 语言实现，不依赖额外的存储配置组件

Traefik 和 Nginx 性能比较

https://www.loggly.com/blog/benchmarking-5-popular-load-balancers-nginx-haproxy-envoy-traefik-and-alb/

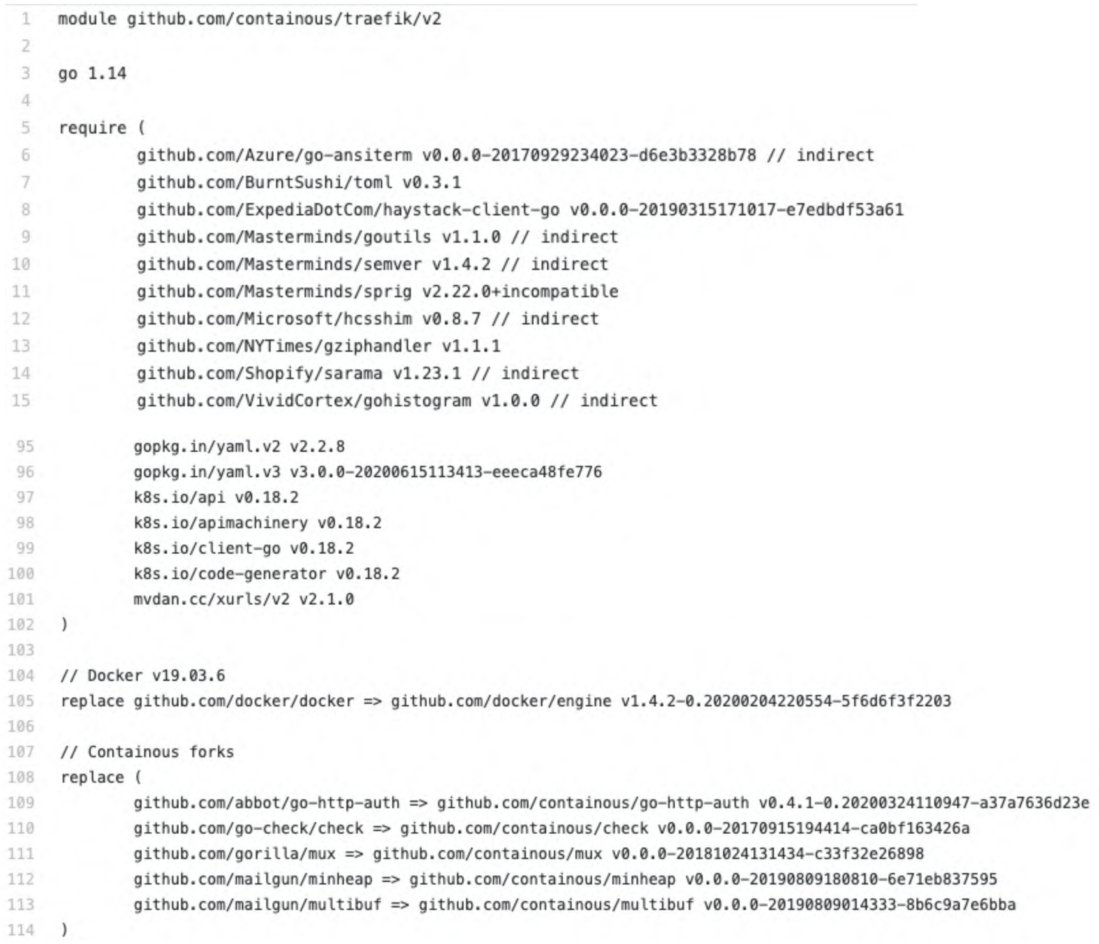


Traefik 和 Ingress-Nignx 组件比较



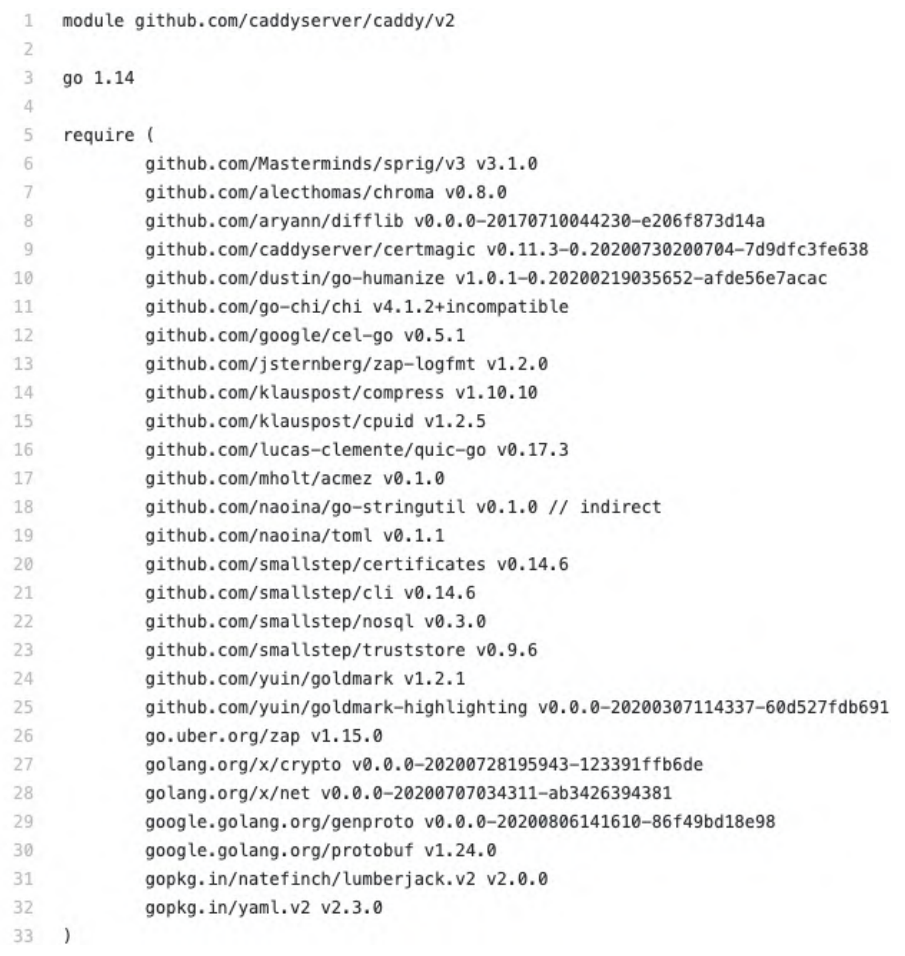
Traefik 的go.mod

traefik star:30k issues:535



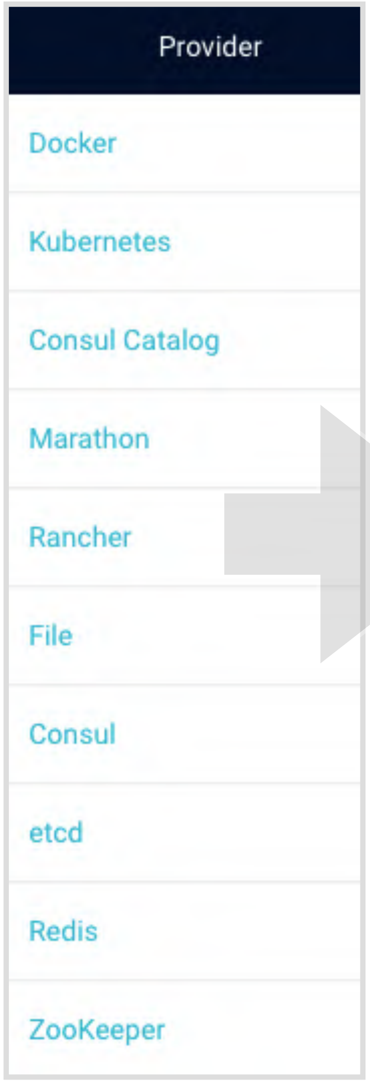
```
module github.com/containous/traefik/v2

go 1.14

require (
        github.com/Azure/go-ansiterm v0.0.0-20170929234023-d6e3b3328b78 // indirect
        github.com/BurntSushi/toml v0.3.1
        github.com/ExpediaDotCom/haystack-client-go v0.0.0-20190315171017-e7edbdf53a61
        github.com/Masterminds/goutils v1.1.0 // indirect
        github.com/Masterminds/semver v1.4.2 // indirect
        github.com/Masterminds/sprig v2.22.0+incompatible
        github.com/Microsoft/hcsshim v0.8.7 // indirect
        github.com/NYTimes/gziphandler v1.1.1
        github.com/Shopify/sarama v1.23.1 // indirect
        github.com/VividCortex/gohistogram v1.0.0 // indirect

        gopkg.in/yaml.v2 v2.2.8
        gopkg.in/yaml.v3 v3.0.0-20200615113413-eeeca48fe776
        k8s.io/api v0.18.2
        k8s.io/apimachinery v0.18.2
        k8s.io/client-go v0.18.2
        k8s.io/code-generator v0.18.2
        mvdan.cc/xurls/v2 v2.1.0
)

// Docker v19.03.6
replace github.com/docker/docker => github.com/docker/engine v1.4.2-0.20200204220554-5f6d6f3f2203

// Containous forks
replace (
        github.com/abbot/go-http-auth => github.com/containous/go-http-auth v0.4.1-0.20200324110947-a37a7636d23e
        github.com/go-check/check => github.com/containous/check v0.0.0-20170915194414-ca0bf163426a
        github.com/gorilla/mux => github.com/containous/mux v0.0.0-20181024131434-c33f32e26898
        github.com/mailgun/minheap => github.com/containous/minheap v0.0.0-20190809180810-6e71eb837595
        github.com/mailgun/multibuf => github.com/containous/multibuf v0.0.0-20190809014333-8b6c9a7e6bba
)
```

caddy star:29.6k issues:71

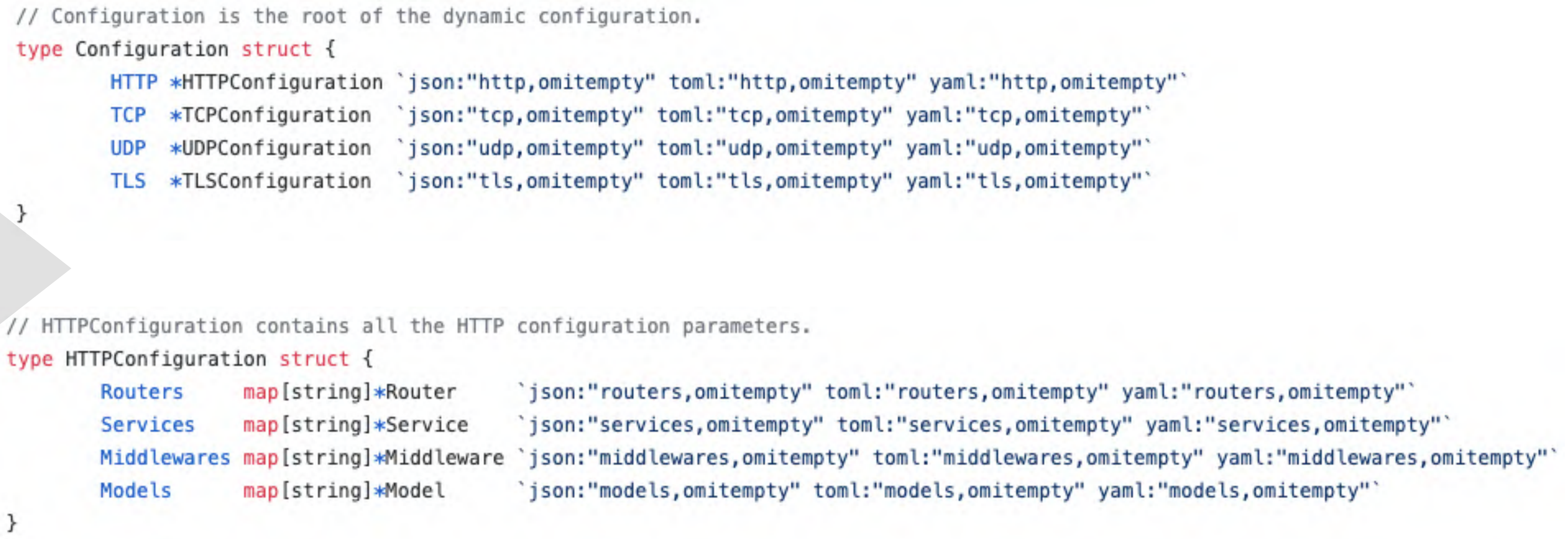


```
module github.com/caddyserver/caddy/v2

go 1.14

require (
        github.com/Masterminds/sprig/v3 v3.1.0
        github.com/alecthomas/chroma v0.8.0
        github.com/aryann/difflib v0.0.0-20170710044230-e206f873d14a
        github.com/caddyserver/certmagic v0.11.3-0.20200730200704-7d9dfc3fe638
        github.com/dustin/go-humanize v1.0.1-0.20200219035652-afde56e7acac
        github.com/go-chi/chi v4.1.2+incompatible
        github.com/google/cel-go v0.5.1
        github.com/jsternberg/zap-logfmt v1.2.0
        github.com/klauspost/compress v1.10.10
        github.com/klauspost/cpuid v1.2.5
        github.com/lucas-clemente/quic-go v0.17.3
        github.com/mholt/acmez v0.1.0
        github.com/naoina/go-stringutil v0.1.0 // indirect
        github.com/naoina/toml v0.1.1
        github.com/smallstep/certificates v0.14.6
        github.com/smallstep/cli v0.14.6
        github.com/smallstep/nosql v0.3.0
        github.com/smallstep/truststore v0.9.6
        github.com/yuin/goldmark v1.2.1
        github.com/yuin/goldmark-highlighting v0.0.0-20200307114337-60d527fdb691
        go.uber.org/zap v1.15.0
        golang.org/x/crypto v0.0.0-20200728195943-123391ffb6de
        golang.org/x/net v0.0.0-20200707034311-ab3426394381
        google.golang.org/genproto v0.0.0-20200806141610-86f49bd18e98
        google.golang.org/protobuf v1.24.0
        gopkg.in/natefinch/lumberjack.v2 v2.0.0
        gopkg.in/yaml.v2 v2.3.0
)
```

Traefik 多配置的实现



Provider
Docker
Kubernetes
Consul Catalog
Marathon
Rancher
File
Consul
etcd
Redis
ZooKeeper



```go
// Configuration is the root of the dynamic configuration.
type Configuration struct {
	HTTP *HTTPConfiguration `json:"http,omitempty" toml:"http,omitempty" yaml:"http,omitempty"`
	TCP  *TCPConfiguration  `json:"tcp,omitempty" toml:"tcp,omitempty" yaml:"tcp,omitempty"`
	UDP  *UDPConfiguration  `json:"udp,omitempty" toml:"udp,omitempty" yaml:"udp,omitempty"`
	TLS  *TLSConfiguration  `json:"tls,omitempty" toml:"tls,omitempty" yaml:"tls,omitempty"`
}
```

```go
// HTTPConfiguration contains all the HTTP configuration parameters.
type HTTPConfiguration struct {
	Routers     map[string]*Router     `json:"routers,omitempty" toml:"routers,omitempty" yaml:"routers,omitempty"`
	Services    map[string]*Service    `json:"services,omitempty" toml:"services,omitempty" yaml:"services,omitempty"`
	Middlewares map[string]*Middleware `json:"middlewares,omitempty" toml:"middlewares,omitempty" yaml:"middlewares,omitempty"`
	Models      map[string]*Model      `json:"models,omitempty" toml:"models,omitempty" yaml:"models,omitempty"`
}
```

Traefik — k8s 的 ingress 配置 Provider

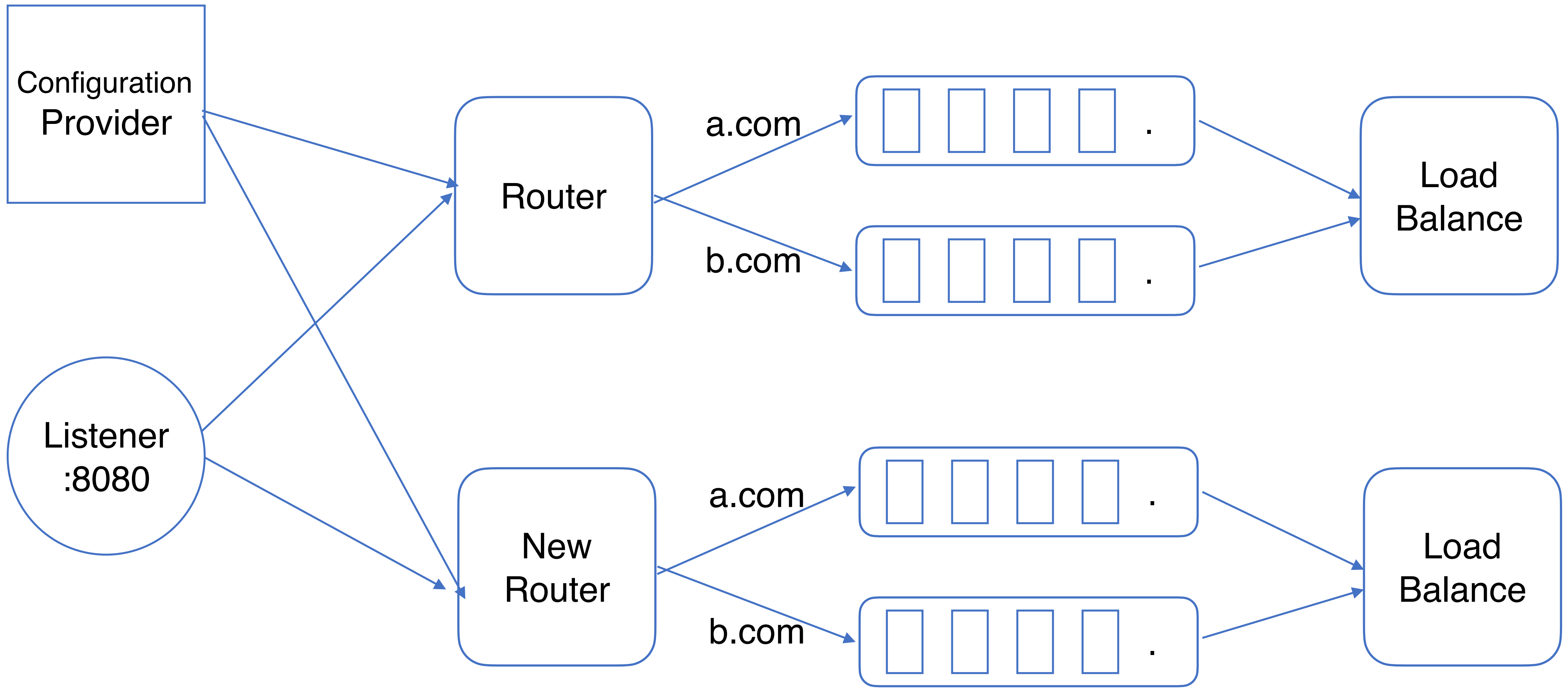
CustomResourceDefinition

- ingressroutes.traefik.containo.us
- ingressservices.traefik.containo.us
- middlewares.traefik.containo.us

- ingressroutetcps.traefik.containo.us
- ingressrouteudps.traefik.containo.us

- tlsoptions.traefik.containo.us
- tlsstores.traefik.containo.us

Traefik 动态配置更新流程



Traefik 中间件

```go
//http.Handler
type Handler interface {
    ServeHTTP(ResponseWriter, *Request)
}

//traefik
func (m *Middleware) ServeHTTP(w http.ResponseWriter, r *http.Request){
    ...
    m.Next(w,r)
    ...
}
```

增加一个类似gin的

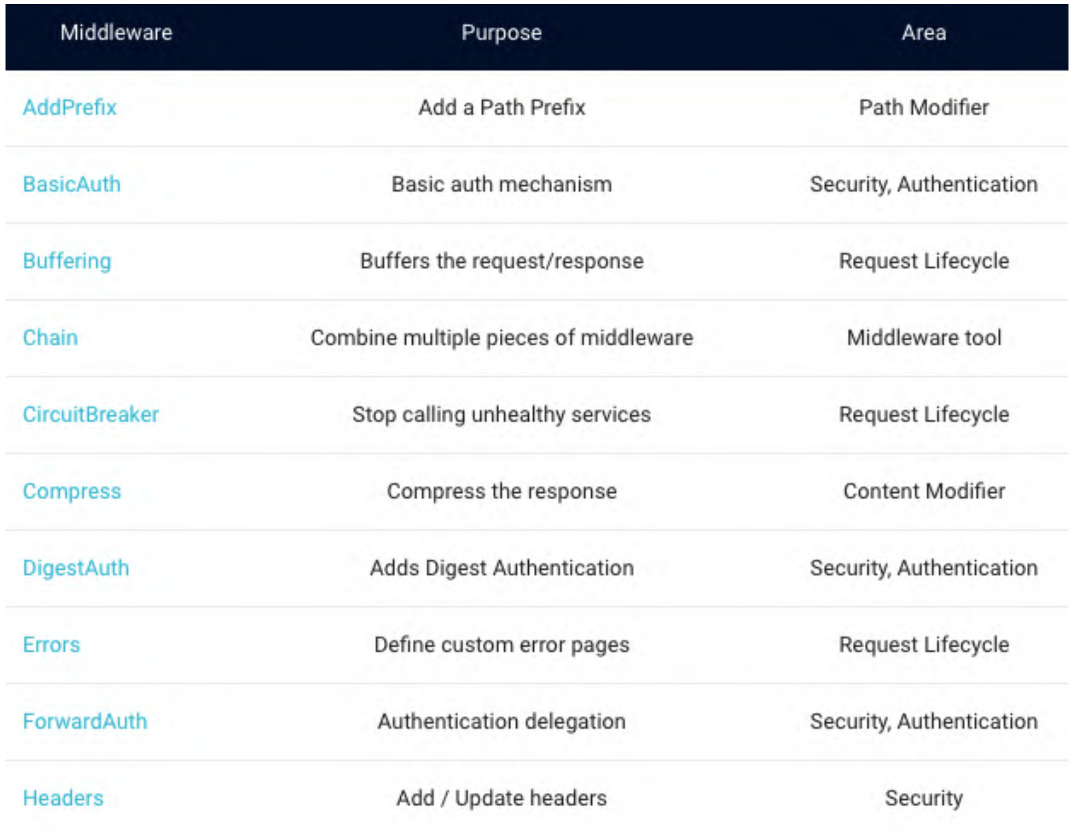
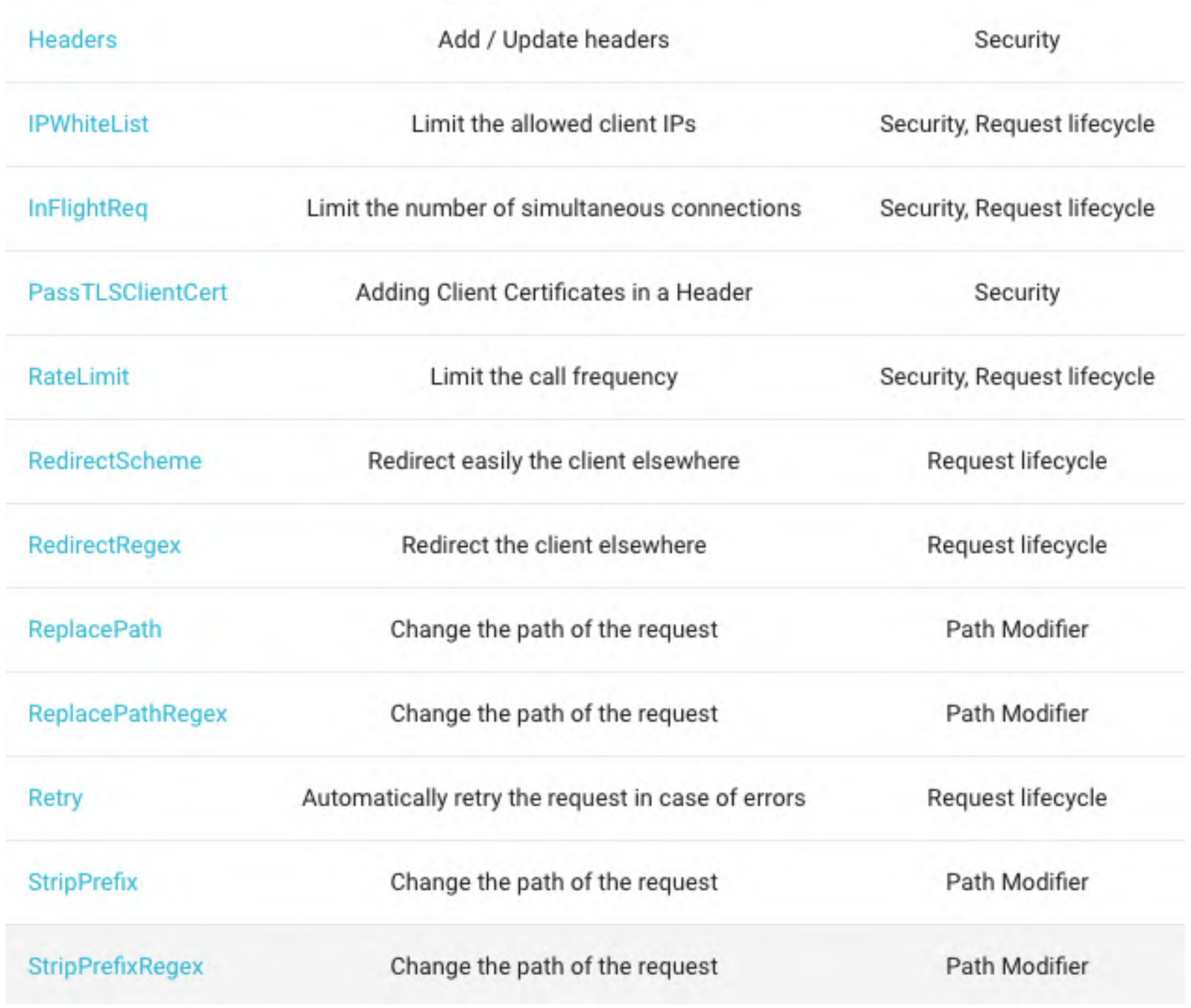
```go
func context.WithValue(parent Context, key, val interface{}) Context
```

```go
// Context is the most important part of gin. It allows us to pass variables between middleware,
// manage the flow, validate the JSON of a request and render a JSON response for example.
type Context struct {
    writermem responseWriter
    Request   *http.Request
    Writer    ResponseWriter
    Params   Params
    ...
```

Traefik 中间件动态加载

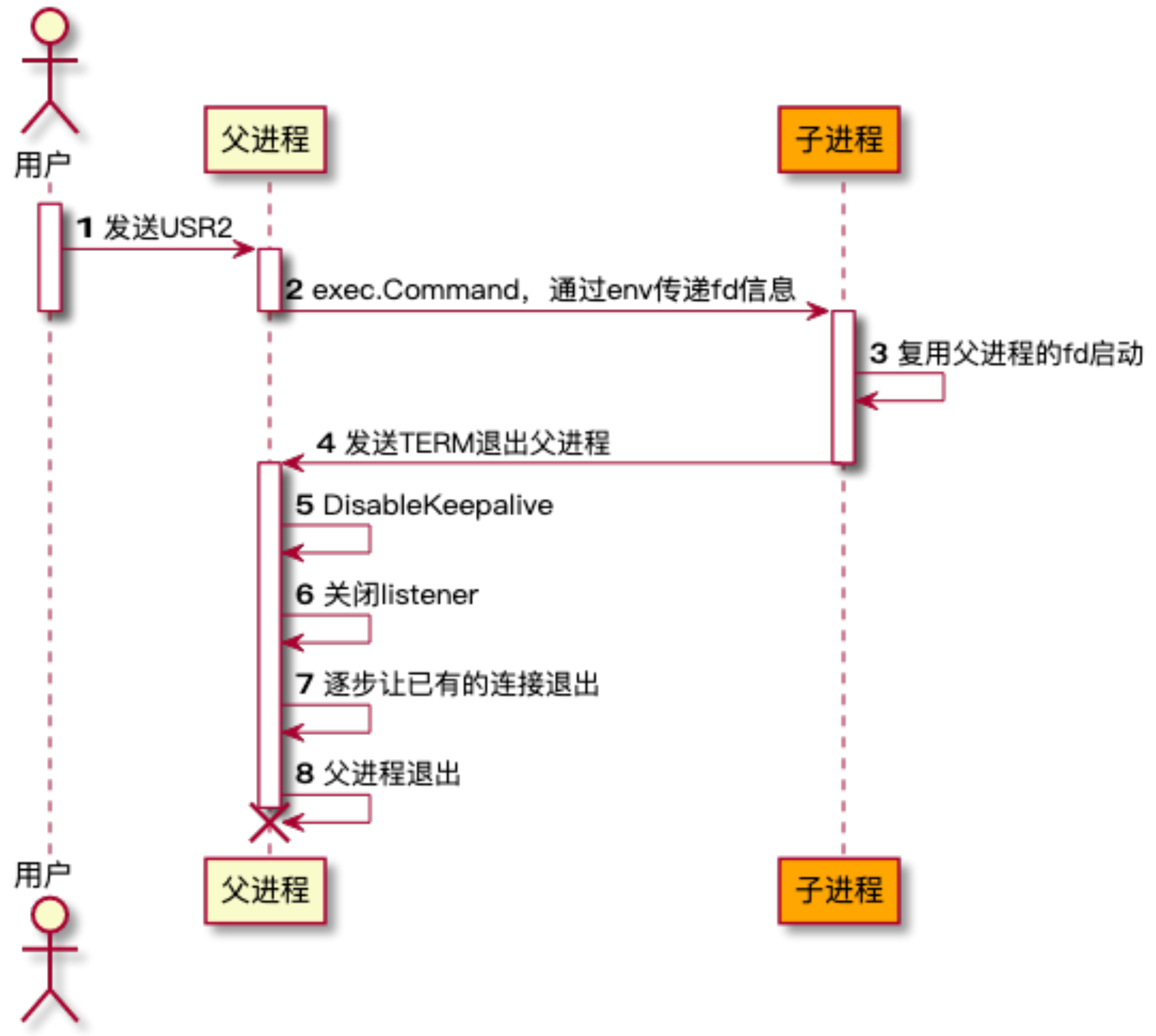
```go
// Config the plugin configuration.
type Config struct {
    // ...
}
// CreateConfig creates the default plugin configuration.
func CreateConfig() *Config {
    return &Config{
        // ...
    }
}
// Example a plugin.
type Example struct {
    next     http.Handler
    name     string
    // ...
}
// New created a new plugin.
func New(ctx context.Context, next http.Handler, config *Config, name string) (http.Handler, err
    // ...
    return &Example{
        // ...
    }, nil
}
func (e *Example) ServeHTTP(rw http.ResponseWriter, req *http.Request) {
    // ...
    e.next.ServeHTTP(rw, req)
}
```

Traefik 提供的中间件

Middleware	Purpose	Area
AddPrefix	Add a Path Prefix	Path Modifier
BasicAuth	Basic auth mechanism	Security, Authentication
Buffering	Buffers the request/response	Request Lifecycle
Chain	Combine multiple pieces of middleware	Middleware tool
CircuitBreaker	Stop calling unhealthy services	Request Lifecycle
Compress	Compress the response	Content Modifier
DigestAuth	Adds Digest Authentication	Security, Authentication
Errors	Define custom error pages	Request Lifecycle
ForwardAuth	Authentication delegation	Security, Authentication
Headers	Add / Update headers	Security
Headers	Add / Update headers	Security
IPWhiteList	Limit the allowed client IPs	Security, Request lifecycle
InFlightReq	Limit the number of simultaneous connections	Security, Request lifecycle
PassTLSClientCert	Adding Client Certificates in a Header	Security
RateLimit	Limit the call frequency	Security, Request lifecycle
RedirectScheme	Redirect easily the client elsewhere	Request lifecycle
RedirectRegex	Redirect the client elsewhere	Request lifecycle
ReplacePath	Change the path of the request	Path Modifier
ReplacePathRegex	Change the path of the request	Path Modifier
Retry	Automatically retry the request in case of errors	Request lifecycle
StripPrefix	Change the path of the request	Path Modifier
StripPrefixRegex	Change the path of the request	Path Modifier

Traefik 热更新二进制文件



```go
cmd := exec.Command(path, args...)
cmd.Stdout = os.Stdout
cmd.Stderr = os.Stderr
cmd.ExtraFiles = fds //111,212
cmd.Env = env //127.0.0.1:2000,:4000
```

Traefik 其它的一些改造

1.Traefik Hash算法跟 OpenResty 不同

2.Traefik 的超时设置

3.Traefik retry 算法

4.Traefik 日志格式

关注又拍云微信公众号，

获取更多干货！

Q & A

又拍云